2012

Information Systems Development Projects as Complex Adaptive Systems

Karlheinz Kautz
Copenhagen Business School, karlheinz.kautz@rmit.edu.au

Information Systems Development Projects as Complex Adaptive Systems

Karlheinz Kautz
School of Information Systems & Technology, University of Wollongong, Wollongong NSW 2522, Australia
email: Kautz@uow.au.edu

Abstract

This research considers information systems development (ISD) projects as complex adaptive systems. We investigate the question whether complex adaptive systems (CAS) theory is relevant as a theoretical foundation for understanding ISD, and if so, which kind of understanding can be achieved by utilizing the theory? We introduce key concepts of CAS theory such as interaction, emergence, interconnected autonomous agents, self-organization, co-evolution, poise at the edge of chaos, time pacing, and poise at the edge of time to analyse and understand ISD in practice. We demonstrate the strength of such a CAS approach through an empirical case study presentation and analysis. While our work contributes to a complexity theory of ISD, the case examination also provides practical advice derived from this perspective to successfully cope with complexity in ISD in an adaptive manner.

Keywords

Information systems development, complexity theory, complex adaptive systems

INTRODUCTION

Contemporary Information Systems Development (ISD) is acknowledged to be a complex activity (see f. ex. Truex et al. 1999, Highsmith 2000, Meso & Jain 2006). Benbya and McKelvey (2006) present a analysis of sources of complexity in contemporary ISD and recognize as such: changing user requirements, changing organizational needs, changing external competitive conditions, increased interdependencies among the involved individuals, organizations and technologies, and the rapid evolution of IS and IT. They state that ISD is often viewed as a complex top-down process and lament that such a perspective falls short in dealing with the identified, but often unexpected contingencies of the ISD process. As an alternative they propose a conception of ISD based on complexity science and theory (Kauffman 1993, Kauffman 1995, Holland 1995, Holland 1998) and propose that ISD projects, following a special branch of complexity science, should be viewed as complex adaptive systems (CAS) and are better understood through the application of CAS. According to complexity science complexity generally refers to an emergent property of systems, which are made of large numbers of self-organizing agents that interact in a dynamic and non-linear fashion and share a path-dependent history (Jacucci et al. 2006 citing Cilliers 1998). CAS theory is a branch of complexity science that studies how a complex system can be adaptive to its environment and how innovative properties of a system emerge from the interactions of its components (Vidgen & Wang 2009). CAS theory has been extensively applied in management and organization studies - Benbaya & McKelvey (2006) and Vidgen & Wang (2009) provide two summaries of this literature - and more recently general writings on the topic in the IS domain (see f. ex. Desai 2005, Jacucci et al. 2006, Merali & McKelvey 2006) as well as specific research on CAS and ISD (Highsmith 2000, Augustine et al. 2005, Benbya & McKelvey 2006, Meso & Jain 2006, Vidgen & Wang 2006, Wang & Vidgen 2007, Kautz & Zumpe 2008, Vidgen & Wang 2009, Wang & Conboy 2009, Kautz & Madsen 2010) have been published. In the discipline of ISD the work has either been conceptual or with a focus on agile software development.

Vidgen and Wang (2009) relate the scarcity of empirical research on CAS in the IS field to the difficulty of turning the abstract concepts of CAS theory into sufficiently concrete case study research. While they provide such a study and a CAS-based framework to investigate and improve the organization of agile software development, they simultaneously criticize the often found simple classifications of agile versus nonagile development practices and call for research, which on a CAS theory background examines closely how development practices are actually implemented beyond such classifications. Kautz and Madsen (2010) provide a similar argument, but also base their research on the objective of understanding agile software development. They suggest that more research is needed to investigate if CAS theory in general or certain CAS concepts in particular are relevant as a theoretical foundation for understanding ISD, and if so, which kind of understanding can be achieved? To answer these questions the research presented here follows Benbya and McKelvey (2006) and considers ISD projects as complex adaptive systems independently of a particular methodological approach chosen for the execution of the project and demonstrates the advantages of such an approach through an empirical case study presentation and analysis. As such it contributes to a complexity theory of ISD and provides practical advice derived from this perspective.

The remainder of the paper is structured as follows. In the next section we present relevant CAS concepts , which, we use to analyze and discuss the case study. Section 3 describes the research approach and section 4 presents an account of the case study that provides contextual information and identifies the applied work practices. In section 5 these practices are discussed in terms of CAS theory and in section 6 we summarize our contributions and our answers to the research questions and point out some limitations of our research approach as well as opportunities for further research.

CAS THEORY - A FRAMEWORK FOR EMERGING ISD PRACTICES

There is no single and fully shared definition of CAS and unifying CAS theory, the following review of the literature on CAS theory as our theoretical background is based on Vidgen and Wang (2009), who have provided a valuable summary on the subject matter, which both takes original CAS literature and its applications in managerial and organizational as well as in IS studies into account. A complex adaptive system consists of a large number of loosely interconnected autonomous parts or agents, each of which behaves according to some set of, sometimes, rather simple rules; these rules require agents to adjust their behavior to that of other agents, whom they interact with, and to adapt to each other; the resulting system behaviors can be very complex (Vidgen & Wang 2009, Wang & Conboy 2009).

Interaction is a significant concept in this context as 'the behavior of the system is determined by the nature of these interactions, not by what is contained within the components. Since the interactions are rich, dynamic, nonlinear, and are fed back, the behavior of the system as a whole cannot be predicted from the inspection of its components. The notion of "emergence" is used to delineate this aspect' (Cilliers 2000). CAS theory rests on the idea that order emerges through the interaction of the agents (Benbya & McKelvey 2006).

Emergence is the property of complex adaptive systems, which creates some greater property of the whole, the system behavior, from the interaction of the parts, the agent behaviors; the emergent system behavior cannot be predicted or fully explained from the measured behaviors of the agents (Highsmith 2000). Interaction and emergence are closely related and link together other generally acknowledged properties of CASs. Beyond the above mentioned interconnected autonomous agents, a number of concepts are frequently used when discussing CASs; these concepts, which we briefly introduce in the following, are: self-organization, co-evolution, the poise at the edge of chaos, time pacing and the poise at the edge of time.

Interconnected autonomous agents, human or non human, have the ability to independently intervene and determine what action to take given their perception of their environment; yet they are interconnected and interact in such a way that they collectively or individually are responsive to change around them, but not overwhelmed by the information flowing to them by this connectivity (Mitleton-Kelly 2003, Vidgen & Wang 2009).

Self-organization then is the capacity of interconnected autonomous agents to evolve into an optimal organized form without external force; it results from the agents' interaction in a disciplined manner within locally defined and followed rules and as such requires a departure from command and control management (Volberda & Levin 2003, Vidgen & Wang 2009).

Co-evolution relates to that a CAS and/or its parts at a sustainable change rate alter their structures and behaviors in response to the interactions of its parts and to the interaction with other CASs, which co-exist in an ecosystem where adaptation by one system affects the other systems; this leads to further adaptations and reciprocal change where the systems do not evolve individually, but concertedly (Kauffman 1993, Vidgen & Wang 2009).

Poise at the edge of chaos describes the ability of a CAS to be at the same time stable and unstable, to never quite lock into place, yet never quite fall apart; the edge is the place, which provides both the stimulation and freedom to experiment and to adapt as well as for novelty to appear, but alsofor the sufficient frameworks and structures to avoid disorderly disintegration; this gives a competitive advantage: CASs that are driven to the edge of chaos out-compete those, which are not (Brown & Eisenhardt 1998, Anderson 1999, Stacy 2003, McMillian 2004, Vidgen & Wang 2009).

Time pacing in this context indicates that a CAS creates an internal rhythm that drives the momentum of change and that change in CASs is triggered by the passage of time rather than the occurrence of events; this stops them from changing too often or too quickly (Brown & Eisenhardt 1998, Vidgen & Wang 2009).

Finally, **poise at the edge of time** conceptualizes a CAS's attribute of simultaneously being rooted in the present, yet aware of the future and its balance and synchronization between engaging enough exploitation of existing resources and capabilities to ensure current viability with engagement of enough exploration of new opportunities to ensure future viability (Brown & Eisenhardt 1998, Volberda & Levin 2003, Vidgen & Wang 2009).

All these core concepts are heavily intertwined and mutually reinforcing (Vidgen & Wang 2006). In summary, complex adaptive systems thus can be characterized through the emergence of co-evolutionary, self-organized behavior, structure and order through the interaction of interconnected autonomous agents in a time-paced rhythm balancing at the edge of time. At the heart of CAS theory's relevance for ISD is the concept of emergence (Highsmith 2000); it appears in relation to all key concepts as:

(1) emergent order resulting from self-organizing agent interactions (Kauffman 1993, Holland 1995)
(2) the emergence of team learning as a result of the interaction and capabilities of interconnected autonomous agents (Mitleton-Kelly 2003)
(3) truly emergent self-organization in contrast to a premeditated or deliberately implemented one by management (Stacey 2003)
(4) co-evolutionary based emergent behavior and structure (Benbya & McKelvey 2006) or even emergent co-evolution (Kelly 1994)
(5) the region of emergent complexity, the edge of chaos (Benbya & McKelvey 2006)
(6) rhythm considered an emergent state of working in the context of time pacing (Vidgen & Wang 2009)
(7) emergent balance between exploitation and exploration at the edge of time (Bocanet & Ponsiglione 2012).

RESEARCH APPROACH

Given the limited literature concerning our research topic, understanding ISD from a complex adaptive systems perspective, our investigation is based on an exploratory, qualitative, single case study (Creswell 2003) of a contemporary ISD project. While it is often stated that it is not possible to generalize and certainly not to theorize from a single case study, Walsham (1995) suggests that it is possible to generalize case study findings among others in the form of a contribution of rich insight. Our research is based on an empirical case study of a commercial ISD project in a large German public sector organization, called WaterWorks, performed by a German software company, called IsDev.The research presented is part of a larger project, which aims at understanding ISD in practice and at contributing to sustainable theories of ISD (References omitted). In the context of this paper we focus on complex adaptive systems theory as a viable foundation for such theories. For this purpose, the collected empirical data was revisited and with regard to the key concepts of CAS theory as identified and discussed in the previous section.

The original empirical data for the case study was collected in semi-structured, open-ended interviews, which were conducted by a team of two researchers in a three days period on the development site. The research team performed 12 interviews with 11 individuals – the IsDev project manager was interviewed twice. This included nearly a third of the development team covering project managers and developers with a wide range of experience and skills (for further information on the development team see the next section) and a representative sample of key players and future users in the customer organization. The interviews were tape-recorded and subsequently transcribed. For the qualitative data analysis a software tool was used. The interview data was supplemented with company and project documents such as method, requirements and release descriptions, as well as project plans. The ISD project was organized in two phases. When this study was performed, phase one, which had lasted 12 months, had been successfully closed and phase two had been going on for 4 months. Responding to an inquiry call during our first data analysis the responsible project manager stated that phase two ended 10 months after our study, on time and budget with all parts of the IS being operational. As a part of the analysis we produced an account of our case of ISD in practice, which in an abbreviated form is presented in the next section.

THE CASE SETTING – THE OMS PROJECT

The project under investigation was concerned with the development of an operations management system (OMS) for the WaterWorks of a large German city by the software company IsDev. The system was developed with a web-based graphical user interface and a backend to interface the technical infrastructure as defined by an underlying ERP system. The OMS project was described by both the customer and the development company as a success. At the time of the project IsDev consisted of about 25 employees, 20 of them were developers, and based its development approach on the development method extreme programming (Beck & Fowler 2001), which provides a number of practices and tools to structure the ISD process. The formalized method includes planning techniques for releases and short iterations called planning games, A5 sized user stories and story cards to specify user requirements, onsite customers to support customer-developer communication, daily stand-up meetings for all project members, pair programming, collective ownership, re-factoring of the developed software, as well as continuous integration and testing to develop the software proper. IsDev had extended the method with some project management processes to cater for their larger projects such as an elaborate overall project plan, formal reporting mechanisms and a formal contract based on an initial requirements specification produced by the customer. The overall development process was a two-phased development approach structured

around an initial requirements specification phase and a development phase. Between these two phases a new formal contract based on the original requirements specification was negotiated. In a first 12 months exploration phase prototypes were developed to catch requirements and possible solutions. This led to the development of a comprehensive requirements document by the customer organization and their decision to contract IsDev also for the actual development of the OMS. In this main development phase the project was organized in 4 subprojects to provide IT support ranging from customer management to the maintenance of the sewer system and about 12 IsDev development staff with multiple roles such as project manager, analyst, customer contact, and developer worked onsite in a building owned by WaterWorks. The overall project team also consisted of a varying number of users with at least one user representing each of the subprojects. A sophisticated management structure and project organization was established and implemented. In addition to two project managers, each representing one of the companies, it consisted of one subproject manager, also acting as contact person, from IsDev and one subproject manager, also acting as onsite-customer, from WaterWorks for each individual subproject. Each subproject thus had about 4 developers including the IsDev subproject manager and at least one WaterWorks representative. The development team was composed of highly educated and motivated, young staff, but only the IsDev subproject managers had extensive experience with the applied method and techniques. Their project manager with 5 years of practice with the method was the most experienced team member, but none of them had ever participated in such a large project.

A first software release was provided after 3 months with the others to be delivered every 3-6 months. Each release was organized in iterations of 3-6 weeks duration, at the time of our investigation each subproject had at least gone through 2 iterations. Working software was produced story card by story card; this attracted the WaterWorks subproject managers and structured their participation in the development process. The working software was the measure of progress and the short releases, iterations, and feedback cycles provided the structure for its development. It was presented to the customer on a weekly basis. Documents were also used. The project was performed based on an elaborate project plan and an overarching requirements specification and for each iteration a number of different, but short and concise documents such as requirement lists and story cards were outlined. Plans and planning structured the ISD process as well. The project had an overall long term project plan developed by the IsDev project manager together with the WaterWorks project manager covering a 14 months period anticipating 3-6 releases depending on the subprojects. A more fine-grained plan was developed for the individual iterations, which made up a release, detailed to single weeks. In the beginning of each iteration the development teams produced and estimated all story cards in team sessions in a planning game. The IsDev staff did most of the work in these games as they - based on the customers' explanations and descriptions - developed the requirements lists and the story cards, but the customer representatives were involved with a decision mandate in the prioritization of the story cards. The planning game and the story cards then offered the devices to perform further planning on the most detailed level for very short periods of time when implementing a story card.The planning games and the stand up meetings were also used to structure the distribution of work tasks. The developers either chose tasks themselves or in common consent were assigned tasks – sometimes on the spot – by their subproject manager without any involvement of the project manager. Stand-up meetings where all teams participated and informed each other about the status of their work, about found solutions and unresolved problems, took place every day. These sessions were originally quite detailed and long, but this practice was later refined and shorter meetings distinguishing between the overall project and the subprojects were introduced and were acknowledged as very helpful.

Pair programming organized the individuals of the development teams to work in shifting pairs of developers in front of a screen while implementing the requirements written down as user stories on story cards as executable code. The developer teams emerged through spontaneous formations. Two mechanisms here were important: to regularly shift a partner and to regularly shift possession of the keyboard within a team. The developers found it difficult to find the appropriate synchronization points at which to change a partner in the teams of 4 developers. No common practice existed. However they did not follow an overly bureaucratic rule such as shifting partner every morning regardless of the status of a story card. To avoid both too much red tape and too much disorder some developers preferred to stay with a partner until a card was closed. However a subproject leader might intervene if a pair had worked together for too long, say 3 days. The developers started out with a practice that one developer exclusively held the keyboard and programmed, while the other watched and sometimes commented. To avoid such situations a process was introduced where using a stopwatch after 20 minutes the keyboard had to switch. This was however abandoned as too bureaucratic and not fruitful in the work environment. Subsequently, the teams found their own pattern.

Dealing with change was an integral part of the daily activities. Beyond the scheduled meetings some of the WaterWorks subproject managers turned up at the project nearly on a daily basis while others came regularly and looked over the shoulders of the developers. Different feedback mechanisms were introduced and provided structures for collecting ideas and change requests. Change requests were brought forward by the customers and users on a short time scale via weekly and bi-weekly feedback sessions and scheduled weekly meetings for all subproject managers built into an iteration based on presentations and demonstrations of working software. The

customer representatives also regularly performed 'road shows' in the user departments to collect feedback and ideas and proposals for improvements. Finally change requests were detected and collected through the scheduled acceptance test sessions for an iteration or a release with customer representatives. Change requests were as a rule dealt with and implemented in the next iteration(s) with the exception of minor changes, which after negotiations and mutual agreement, were implemented as part of the current iteration.

THE OMS PROJECT AS A COMPLEX ADAPTIVE SYSTEM

Interactions are a significant characteristic of a CAS. Within the overall project organization, which had been established by the management of the two project partners and which consisted of a two phased development approach, an initial project plan, four subprojects, a formal contract, and a first requirements specification, interactions in the OMS project between the project participants and with other WaterWorks staff took place throughout the whole project in multiple forms, formal and informal, and at numerous occasions, organized and spontaneous. They could be found during the preparations of an iteration in the planning games, in the scheduled weekly subproject manager meetings, the bi-weekly and weekly feedback sessions, during development when clarifying urgent issues face-to-face, on the telephone or via email and through the almost daily informal visits and while watching the developers' work, as well as during other presentations and evaluations of the working software in trials and the acceptance tests. Intensive interaction between the IsDev members of the development teams also took place during the development of the requirements list and story cards, the daily stand-up meetings and during the pair programming. This is ample evidence of the rich, intense, dynamic, and non-linear interactions, which were fed back into the project and determined its performance; it shows that its course could not be predicted. The concept of emergence describes this characteristic of CASs. In the following we will revisit the OMS project and accentuate the different facets of emergence and key notions of CAS theory to provide a better understanding of contemporary ISD and to derive some insight for the organization of promising ISD projects.

Interconnected, Autonomous Agents and the Emergence of Team Learning

In the OMS project both the IsDev and the WaterWorks staff acted as autonomous, interconnected agents. Their autonomy was expressed in numerous ways. The autonomy of the WaterWork staff in the project showed when they took part in prioritizing, estimating, designing, and especially approving design decisions where the onsite customers had a mandate to make decisions and exerted power. Autonomy also became apparent in the liberty the developers had when distributing and picking tasks and when implementing the story cards, the developers acted largely self-governing with regard to the design decisions they made. Finally, the developers were sovereign when choosing a partner for the pair programming. Still the project participants were highly interlinked and maintained their relationships through the above described structures and measures, which supported various forms of interactions to achieve interconnectivity across and within the four development teams. As also presented by Vidgen and Wang (2009), in general encouraged by the physical presence and closeness in the project facilities onsite at the customer premises, the sharing of project-relevant knowledge in the scheduled weekly feedback sessions and daily stand-up meetings in combination with all team members' involvement in project management, in design decisions and in all development activities, as well as their self-assignment of tasks based on competence and interest, and the rotation of direct collaboration partners in the programming pairs lead to the emergence of learning of the entire team.

Emergent Self-Organization and the Emergence of Order

The concept of self-organization is closely related to the concept of interconnected, autonomous agents (Volberda & Levin 2003, Vidgen & Wang 2009). It is thus not unexpected that the same structures and mechanisms such as short communication paths, shared responsibility for governance and decision making, involvement in all activities as well as task self-assignment were prominent in the support of self-organization. The described development of how the practices of stand-up meetings and pair programming were first implemented and later refined illustrates the emergence of self-organization of autonomous project members and the subsequent emergence of order in the development process of the OMS project and shows that individual and team discipline are not in conflict with, but a vital element of self-organization. The role of the IsDev subproject project managers also deserves some attention. With largely the same development tasks and their sharing of responsibility they appeared more like peers and a part of the team, who primarily, instead of controlling, acted as facilitators and created an environment that fostered self-organization. In this context the way feedback, beyond the self-organized regular feedback sessions, was provided and gathered from the customer staff and subsequently handled in orderly form as part of the planning activities also reflects the self-organization of the project members where some customers gave their comments in passing, others by phone, and yet others per e-mail. Finally, the IsDev staff members' visits to some departments are further examples of self-organization.

Poise at the Edge of Chaos and the Region of Emergent Complexity

The OMS project was continuously in a state of bounded instability, which means that it – paradoxically – was simultaneously stable and unstable (Stacey 2003). The two-phased development approach, the formal contract, the initial project plan and the first requirements specification as well as the overall implemented project organization acted as superordinate structuring mechanisms that created a relative stable space within which the development process and the working software could unfold. Beyond the direct reaction on manageable clarifications and refinements of existing requirements, the handling of the constant requests for changes and new requirements illustrates how the project maneuvered in a region of emergent complexity and balanced at the edge of chaos. The short iterations with frozen requirements, the frequent planning including the estimation of tasks with regard to the developers' capabilities and the prioritization of tasks with regard to the actual situation and not least the dirigible size of the individual tasks and user stories, whose descriptions could exceed not an A5 story card, as well as the stand-up meeting and feedback sessions all allowed for the flexible handling of the large amount of new, incoming demands and ideas and at the same time provided a frame for stability as they structured the project participants' day-to-day activities and helped them to know what to do, and when, and what to expect from others.

Emergent Co-evolution and the Emergence of Behavior and Structure

Co-evolution emerged in the OMS project through the above described multiple forms of close interactions in which the IsDev and the WaterWorks staff shared knowledge and learned from each other. The mutual learning had the reciprocal effect of reinforcing the emerging structures of collaboration and interaction as well as the behavior of the individual project participants. In particular the continuous gathering and refining of requirements, the frequent presentations and delivery of working software fuelled this process and provided the customers with opportunities to learn how to use the OMS and to create new ideas, which were fed back to the development team and became part of the working software. Thus, also the working software co-evolved with the human actors. Together, this demonstrates the co-evolution of both, people, processes and products and unpredictable emergent behavior of the different entities of the CAS during ISD (Meso & Jain 2006).

Time Pacing and the Emergence of Rhythm

In the OMS project, the two-phased development approach and the overall project plan set the time frame for the development process. The frequent releases and in particular the iterations, which - although of varying duration - were comparably short and of a pre-defined length, the formal weekly and bi-weekly meetings and feedback sessions, as well as the daily meetings were performed continuously in accordance with an internally set pace. Based on regular planning and, if necessary, rescheduling with regard to releases, iterations and daily activities this provided the appropriate intervals to match and handle the changes, which were brought up continuously during the project. Together with the manageable size of the requirements on the story cards it supported the emergence of a lasting working rhythm, which becomes most obvious in the way the pair programming both with regard to shifting partners and keyboard developed. This way, time pacing as also reported by Vidgen and Wang (2006) as an organizing principle created an internal rhythm, which drove change in the project in accordance with the passage of time and which at the same time allowed for flexibility (Vidgen & Wang 2009).

The Edge of Time and the Emergent Balance of Exploitation and Exploration

In the OMS project the center of attention was always the current iteration and the current user stories while taking the existing working software and the design for future extensions into account. The iterations of using the existing requirement lists to produce story cards and write working software whilst investigating prospective options with information from and in consultation with customer staff, with its root in the presence and simultaneous awareness of the past and the future, balanced the concurrent exploitation of existing knowledge and the exploration of new knowledge. This has been characterized as the edge of time by Brown and Eisenhardt (1998). The daily stand-up meetings and the pair programming served the purpose of keeping a focus on today at the same time as preparing for the future. This is also true for the frequent presentations of working software, while the overall project plan, as well as the frequent planning sessions, which were structured around releases, iterations, planning games, and the implementation of working software supported a focus on and constituted a manifestation of both the past and the future. Beyond product-oriented exploitation and exploration and providing the opportunity for direct reaction, the frequent manager meetings and feedback sessions were also used by the project participants to think about their own behavior and to review and improve the development process. This is reflected in the flexible, but for an iteration fixed, time boxes and the developed formats for the stand-up meetings and the pair programming.

DISCUSSION

CAS theory has been recognized as a valuable approach to understand contemporary ISD and several research teams have applied the theory and provide organizing principles and suggestions for best practice for the ISD process. Benbya and McKelvey (2006) present some conceptual work. They focus on the co-evolutionary aspect of CAS and put forward seven sometimes rather abstract principles of adaptive success and discuss their bearings on ISD problems. Consequently their advice is quite abstract and in parts not easy to implement. Among the principles are the principle of adaptive tension, which relates to the dynamic nature of the development process and the tensions caused by contradictory requirements, the principle of requisite complexity, which relates to the unpredictable nature of future requirements and the principle of causal intricacy, which relates to the dynamic and continually shifting links between institutional, business and technical requirements.

Meso and Jain (2006) identify the seven principles of CAS, which to some extent overlap with those described by Benbya and McKelvey (2006), but whose meanings are more intuitive and easier to grasp. They are the principles of open systems, of interactions and relationships, of transformative feedback loops, of emergent behavior, of distributed control, of shallow structure, and of growth and evolution. The authors provide a largely conceptual argument, which they support with various, partly unrelated, examples from the literature. They map the principles to agile development practices and from this mapping derive more concrete, yet generic recommendations for best practices for ISD. Their recommendations are: (1) Let actual solutions, processes, work patterns, team configurations and interactions emerge naturally. (2) Do minimal planning; limit emphasis on documentation. (3) Allow for manageable experimentation in product design and with. (4) Develop, test and validate the IS iteratively and compare with past solutions. (5) Allow solutions to be responsive to emerging changes. (6) Strive for componentization. (7) Allow the development process to be determined by local needs. (8) Use an iterative, time-boxed, and modular development process. (9) Reevaluate the development process frequently. (10) Delegate responsibility to local development units. (11) Involve stakeholders and fellow developers. (12) Reevaluate team configuration frequently. Vidgen and Vang (2009), with a focus on co-evolution and agile development, develop a framework, which is grounded in the work of Volberda and Levin (2003) on coevolving self-organizing organizations. They summarize the key concepts of CAS in three guiding principles for the organization of agile software development as matching co-evolutionary change rate, optimizing self-organization, and synchronizing concurrent exploitation and exploration. Based on an actual and sound empirical case they identify six capabilities of agile teams, namely co-evolution of IT team and customer to create business value, sustainable working with rhythm, collective mindfulness, sharing and team learning, process adaptation and improvement, and product innovation. They find practices similar to those made out by Meso and Rain (2006), but consider them as enablers of agility. Correspondingly, they also present inhibitors for agility.

Our work is based on these predecessors, but rather than organizing principles or focusing on particular concepts, we chose as a starting point for our analysis a set of concepts, which characterize CASs. We provide a coherent empirical account of a contemporary ISD project. Our investigation shows how these concepts are interrelated. The practices, which we found on this background, are of the same kind than those depicted by Meso and Jain (2006) and Vidgen and Wang (2009), and in line with these authors we consider them decisive and recommend them. However, beyond confirming many of the earlier presented findings, we extend the line of reasoning and argue that rather than being enablers for agility these practices are properties of complex, adaptive systems to cope with complexity and to be adaptive. If ISD is understood as CAS certain characteristics of the process facilitate good performance while others inhibit it. We contend that they are not just limited to and important for agile development methods, but from a CAS perspective, in general for contemporary ISD, which has to deal with turbulent and changing environments.

Our focus is on the ISD process as such and not on a particular approach or method, which has to be adapted – in contrast to work which takes as starting point an agile method and asks how it can be adapted in different contexts (Cao et al. 2009). We have thus so far not particularly emphasized that the development approach of the OMS project was based on a combination of plan-based and agile practices. CAS theory sheds new light on the discussion concerning the advantages and disadvantages of mixing plan-driven and agile approaches, methods and practices (Abrahamsson et al. 2003, Boehm & Turner 2004, Port & Bui 2009) and goes beyond actual problems and practices as are currently coined in the literature to form a deeper theoretical understanding of ISD (Kautz et al. 2007): While such an integration based on such simple classifications from a pragmatic perspective might be important and is supported by CAS, it seems to be based on a false dichotomic split between agile and other methods (Boehm & Turner 2003) and appears to be theoretically secondary and less relevant. It might even render the plan-based versus agile approaches debate as an ‘either-or’ matter redundant (Austin & Devin 2009) as long as the chosen approach allows for organizing for adaptation and adaptive ISD based on a set of simple foundational rules. In the OMS project the overall plans and requirements documents and the possibility to change, revise, and refine these plans and requirements create conditions in which adaptive activities can

flourish; as such the OMS project was not plan-driven, but planning driven (Vidgen & Wang 2009). In line with Vidgen and Wang (2009) we find that recurrent planning is a consequence of frequent feedback due to the close relationships and the interactions between all those involved in the project and that both high-level, sketchy, long-term plan and detailed, accurate, short-term plans are necessary and beneficial in such a process.

Vidgen and Wang (2009) in the context of self-organization also identify collective mindfulness as a capability of an agile team and Matook and Kautz (2008) demonstrate the relationship between individual and collective mindfulness and agile development and show how mindful behaviour may contribute to successful ISD. Butler and Gray (2006) suggest that individual and collective mindfulness increase organizations' ability to perform in problem solving situations and dynamic, unstable environments such as ISD. According to these authors "Collective mindfulness requires organizations to couple the ability to quickly detect issues, problems, or opportunities with the power to make organizationally significant decisions. This may be accomplished by moving decision-making authority or creating an organizational environment that enables the smooth interaction of perception and action." These are characteristics, which are also associated with ISD when understood as CAS as we demonstrated with our case. In collective mindfulness, existing expectations are continuously scrutinized and refined according to new experiences in order to be able to invent new expectations for dealing with unprecedented situations to improve foresight and current functioning (Weick & Sutcliffe 2001). There are five key aspects, which characterize collective mindfulness. These are preoccupation with failure, reluctance to simplify, attention to operations, commitment to resilience, and migration of decisions to expertise (Weick & Sutcliffe 2001). At the individual level Langer (1997) also distinguishes five constituent components of mindfulness: openness to novelty, alertness to distinction, sensitivity to different contexts, awareness of multiple perspectives and orientation in the present. As an example in the OMS project the behavior that led to the emergence of rhythm in the stand-up meetings and the pair programming in accord with self-organization and autonomy while balancing at the edge of chaos and time exhibits both characteristics of collective and individual mindfulness in the form of alertness to distinction, sensitivity to different contexts, awareness of multiple perspectives, orientation to the present as well a willingness to learn from errors, an aspiration to perceive problems from different perspectives, an ability to cope with problems and the migration of decision to expertise. Thus, we provide further empirical evidence on the role of mindfulness in coping with ISD as CASs, which can be extended beyond its relationship to self-organization to other concepts of CAS theory. We also contend that mindfulness further than agile development is a valuable concept to understand and organize contemporary ISD in general.

CONCLUSION

In this paper we start out with the premise that contemporary ISD is a complex activity. On this background we investigated the question whether CAS theory is relevant as a theoretical foundation for understanding ISD, and if so, which kind of understanding can be achieved by utilizing the theory? By doing so, we followed Baskerville et al. (2011), who argue that in a post-agility era of ISD further theories are needed to understand contemporary ISD. On this basis we demonstrate the strength of such a CAS approach and show that CAS is useful as it directs the focus on characteristics and practices of the development process, which by organizing and facilitating for adaptation, might lead to thriving ISD projects. Grounding explanations in CAS theory, our presentation and examination of an empirical case study provide an argument why and how, despite some identified challenges, project staff successfully carries out an ISD project. Our investigation also illustrates that CAS theory is valuable not just for understanding the so-called agile development methodologies and agile development, but independently of a particular methodological approach for the execution of a project, more general for comprehending contemporary ISD. Understood as CAS certain characteristics of the ISD process assist good performance while others impede it. We contend that these organizing principles, practices and capabilities are not just limited to agile development methods, but from a CAS perspective, are valid in general for contemporary ISD, which has to deal with turbulent and changing environments. Agile development is only one way of dealing with present day ISD. We therefore argue that when studying and discussing ISD to gain in-depth insight it is not that relevant to look if an applied practice belongs to a certain, agile or traditional, 'school' or a combination of these two, but how it is actually used in the particular case.

In conclusion, we recognize that our research is based on a single case study. While it is possible to generalize case study findings in the form of a contribution to rich insight (Walsham 1995), further claims concerning generalizability of the findings cannot be made; as an exploratory study our research provides a foundation for establishing a complexity theory as a viable theory of ISD, and for further research efforts needed concerning such a theory. In this context the above briefly discussed relationship between complexity and mindfulness in ISD deserves attention in future research. Another open question is the issue of funding processes and the impact of pricing structures on ISD practice in relation to control and flexibility in environments, which acknowledge complexity. While they have been discussed and studied in the context of agile development methods (Highsmith 2002, Cao et al. 2012), they have not yet been investigated from a general complexity theory perspective on ISD and provide another worthwhile avenue for further research.

REFERENCES

Anderson, P. 1999. "Complexity Theory and Organization Science," *Organization Science (10:3), pp.* 216–232.

Abrahamsson, P., Warsta, J., Siponen, M. T. and Ronkainen, J. 2003. "New Directions on Agile Methods: A Comparative Analysis", in *Proceedings of the 25th IEEE International Conference on Software Engineering*, Portland, Oregon, pp. 244-254.

Augustine, S., Payne, B., Sencindiver, F., and Woodcock, S. 2005. "Agile project management: steering from the edges," *Communications of the ACM (48:12)*, pp. 85-89.

Austin, R. D., and Devin, L. 2009. "Research Commentary—Weighing the Benefits and Costs of Flexibility in Making Software: Toward a Contingency Theory of the Determinants of Development Process Design," *Information Systems Research* (20:2), pp. 462-477.

Baskerville, R., Pries-Heje, J., Madsen, S. 2011. "Post-agility: What follows a decade of agility?," *Information and Software Technology (53)*, pp. 543-555.

Beck, K., and Fowler, M. 2001. *Planning Extreme Programming* (2nd ed.), Boston, MA: Addison-Wesley.

Benbya, H., and McKelvey, B. 2006. "Toward a complexity theory of information systems development," *Information Technology & People (19:1)*, pp. 12-34.

Bocanet, A., and Ponsiglione, C. 2012. "Balancing exploration and exploitation in complex environments," *VINE - The Journal of Information and Knowledge Management Systems* (42:1), pp. 15-35.

Boehm, B., and Turner, R. 2003. "Using Risk to Balance Agile and Plan Driven Methods," *Computer (36:6)*, pp. 57-66.

Boehm, B., and Turner, R. 2004. "Balancing agility and discipline: evaluating and integrating agile and plan-driven methods," in *Proceedings of the 26th IEEE International Conference on Software Engineering*, Washington DC, pp. 718–719.

Brown, S., and Eisenhardt, K. 1998. *Competing on the Edge: Strategy as Structured Chaos*. Boston, MA: Harvard Business School Press.

Butler, B. S., and Gray, P. H. 2006. "Reliability, Mindfulness and Information Systems", *MIS Quarterly (30:2)*, pp 211-224.

Cao, L., Mohan, K., Xu, P., and Ramesh, B. 2009. "A framework for adapting agile development methodologies," *European Journal of Information Systems (18:4)*, pp. 332-343.

Cao, L., Mohan, K., Ramesh, B., and Sarkar, S. 2012. "Adapting funding processes for agile IT projects: an empricial inverstigation," *European Journal of Information Systems (21:1)*, pp. 1-15.

Cilliers, P. 1998. *Complexity and Postmoderism: Understanding Complex Systems*, London, UK: Routledge.

Cilliers, P. 2000. "What Can We Learn From a Theory of Complexity?," *Emergence (2:1)*, pp. 23-33.

Creswell, J.W. 2003. Research *design - Qualitative, Quantitative and Mixed Methods Approaches*, Thousand Oak, CA: Sage Publications.

Desai, A. 2005. "Introduction into special section on Adaptive Complex Enterprises," *Communications of the ACM* (48:5), pp. 32-35.

Highsmith, J. 2000. Adaptive *Software Development: A Collaborative Approach to Managing Complex Systems*, New York, USA : Dorset House Publishing.

Highsmith, J. 2002. *Agile Software Development Ecosystems*, Boston, MA, USA: Addison-Wesley, Pearson Education.

Holland, J.H. 1995. *Hidden Order: How Adaptation Builds Complexity*, Reading, MA: Addison-Wesley.

Holland, J.H. 1998. *Emergence: From Chaos to Order*, Cambridge, MA: Perseus Publishing.

Jacucci, E., Hanseth, O., Lyytinen, K. 2006. "Taking complexity seriously in is research. Introduction to the Special Issue," *Information Technology & People (19:1)*, pp. 5-11.

Kauffman, S. 1993. *The Origins of Order: Self-Organization and Selection in Evolution*, New York, NY: Oxford University Press.

Kauffman, S. 1995. *At Home in the Universe*, New York, NY: Oxford University Press.

Kautz, K., Madsen, S., Nørbjerg, J. 2007. "Persistent Problems and Practices in Information System Development," *Information Systems Journal (17:3)*, pp. 217-239.

Kautz, K., and Zumpe, S. 2008. "Just Enough Structure at the Edge of Chaos: Agile Information Systems Development in Practice," in *Proceedings of the International Conference XP 2008*, Limerick, Ireland. Abrahamsson, P. et al. (eds.), Berlin, Germany: Springer Notes in Business Information Processing, pp. 137-146.

Kautz, K., and Madsen, S. 2010. "Understanding Agile Software Development in Practice," in *Proceedings of the 2010 International Conference on Information Resources Management*, Rose Bay, Jamaica.

Kelly, K. 1994. *Out of Control - The New Biology of Machines, Social Systems, and the Economic World*, New York, NY: Addison-Wesley.

Langer, E.J. 1997. *The Power of Mindful Learning*, Cambridge MA: Perseus Publishing.

Matook, S., and Kautz, K. 2008. "Mindfulness and Agile Software Development," in *Proceedings of the 19th Australasian Conference on Information*, Christchurch, NZ, pp. 638-647.

McMillan, E. 2004. *Complexity, Organizations, and Change*, London, UK: Routledge.

Merali, Y., and McKelvey, B. 2006. "Using complexity science to effect a paradigm shift in information systems for the 21st century," *Journal of Information Technology (*21:4), pp. 211-215.

Meso, P. and Jain, R. 2006. "Agile Software Development: Adaptive Systems Principles and Best Practices," *Information Systems Management (23:3)*, pp.19-30.

Mitleton-Kelly, E. 2003. "Ten principles of complexity and enabling infrastructures," In *Complex systems and evolutionary perspectives on organisations: the application of complexity theory to organisations*. Mitleton-Kelly, E. (ed.) Oxford, UK: Elsevier Science Ltd, pp. 3-20.

Port, D., and Bui, T. 2009. "Simulating mixed agile and plan-based requirements prioritization strategies: proof-of-concept and practical implications," *European Journal of Information Systems* (18:4), pp. 317-331.

Stacey, R. D. 2003. *Strategic Management and Organisational Dynamics: The Challenge of Complexity* (4th ed.), Harlow, UK : Financial Times, Prentice Hall.

Truex, D.P., Baskerville, R., Klein, H. 1999. "Growing Systems in Emgerent Organizations," *Communications of the ACM (42:8)*, pp. 117-123.

Walsham, G. 1995. "Interpretive Case Studies in IS Research: Nature and Method", *European Journal of Information Systems (4:2)*, pp. 74-81.

Wang, X., and Vidgen, R. 2007. "Order and Chaos in Software Development: A comparison of two software development teams in a major IT company," in *Proceedings of the 16th European Conference on Information Systems*, Winter, R., et al. (eds.), St. Gallen, Switzerland , pp. 807-818.

Wang, X., and Conbory, K. 2009. "Understanding Agility in Software Development through a Complex Adaptive Systems Perspective," in *Proceedings of the 17th European Conference on Information Systems*, Verona, Italy.

Weick, K., and Sutcliffe, K. H. 2001. *Managing the Unexpected: Assuring High Performance in an Age of Complexity*, San Francisco, CA: Jossey-Bass.

Vidgen, R., and Wang, X. 2006. "Organizing for Agility: A Complex Adaptive Systems Perspective on Agile Software Development Process," in *Proceedings of the 14th European Conference on Information Systems*, Goeteborg, Sweden. Ljunberg J., and Andersson, M. (eds.), pp. 1316-1327.

Vidgen, R., and Wang, X. 2009. "Coevolving Systems and the Organization of Agile Software Development," *Information Systems Research (20:3*), pp. 355-376.

Volberda, H. W., and Lewin. A. Y. 2003. "Guest editors' introduction coevolutionary dynamics within and between firms: From evolution to co-evolution," *Journal of Management Studies (*40:8), pp. 2111-2136.

COPYRIGHT